CLAIMS

What is claimed is:

1. A method of ensuring the integrity of a software variable stored in a memory, comprising the steps of:

verifying the integrity of a hardware environment for the software variable, wherein the hardware environment comprises a processor, a plurality of registers, and a plurality of storage locations in said memory;

calculating a first copy of said software variable using said processor and a first set of said plurality of registers;

calculating a second copy of said software variable different from said first copy of said software variable using said processor and a second set of said plurality of registers;

storing said first said software variable in a first portion of said verified plurality of storage locations in said memory;

storing said second said software variable in a second portion of said verified plurality of storage locations in said memory; and

comparing said first copy of said software variable from said first portion of said storage locations with said second copy of said software variable from said second portion of said storage locations to thereby ensure the integrity of said software variable stored in said memory.

2. The method of claim 1 further comprising the step of disabling interrupts to said processor prior to said step of calculating said first said software variable.

3. The method of claim 2 further comprising the step of enabling said interrupts after said step of storing said second said software variable.

4. The method of claim 1 wherein said second copy of said software variable is a twos complement of said first copy of said software variable.

5. The method of claim 1 wherein said step of verifying the integrity of a plurality of storage locations in said memory is based at least upon a March C test.

6. The method of claim 5 wherein said March C test is performed in more than one direction.

7. The method of claim 1 wherein said step of verifying the operational integrity of said processor is based at least upon a seed and key test.

8. The method of claim 7 wherein said step of verifying the operational integrity of said processor is based at least upon the use of a redundant processor.

9. The method of claim 1 wherein said step of verifying the integrity of a plurality of registers is based at least upon a checksum test.

10. An apparatus for calculating and storing a software variable in a memory, comprising:

a processor coupled to said memory; and

a plurality of registers coupled to said processor, wherein said processor is configured to verify the integrity of said memory and said plurality of registers, and said processor is further configured to calculate and store said software variable in said memory after said verifying said memory and said plurality of registers; and

wherein said processor is further configured to verify the integrity of said stored software variable over multiple software loops.

11. The apparatus of claim 10 wherein said processor comprises an Arithmetic Logic Unit (ALU).

12. The apparatus of claim 10 wherein said memory is configured as a Random Access Memory (RAM).

13. The apparatus of claim 10 wherein said plurality of registers are configured as a Read Only Memory (ROM).

14. A method of ensuring the integrity of a calculated software variable stored in a memory, comprising the steps of:

verifying the integrity of hardware used to calculate and store said calculated software variable, wherein said hardware comprises said memory;

computing at least two copies of said calculated software variable;

storing the at least two copies of said calculated software variable in separate portions of said memory; and

testing the integrity of said calculated software variable over a plurality of software test loops, wherein the testing step comprises comparing the at least two copies of said calculated software variable to each other.

15. The method of claim 14 wherein said verifying step comprises testing at least those portions of a processor and said memory involved in the calculation and storage of said calculated software variable.

16. The method of claim 15 wherein the verifying step further comprises testing a register used in computing the at least two copies of said calculated software variable.

17. An apparatus for ensuring the integrity of a software variable, comprising:

means for verifying the integrity of hardware used to calculate and store said calculated software variable, wherein said hardware comprises said memory;

means for computing at least two copies of said calculated software variable;

means for storing the at least two copies of said calculated software variable in separate portions of said memory; and

means for testing the integrity of said calculated software variable over a plurality of software test loops, wherein the testing means comprises means for comparing the at least two copies of said calculated software variable to each other.

18. An apparatus for ensuring the integrity of a software variable stored in a memory, comprising:

a first module for verifying the integrity of hardware used to calculate and store said calculated software variable, wherein said hardware comprises said memory;

a second module for computing at least two copies of said calculated software variable;

a third module for storing the at least two copies of said calculated software variable in separate portions of said memory; and

a fourth module for testing the integrity of said calculated software variable over a plurality of software test loops, wherein the testing means comprises means for comparing the at least two copies of said calculated software variable to each other.

19. A digital storage medium having computer-executable instructions stored thereon, the instructions comprising:

a first module for verifying the integrity of hardware used to calculate and store said calculated software variable, wherein said hardware comprises said memory;

a second module for computing at least two copies of said calculated software variable;

a third module for storing the at least two copies of said calculated software variable in separate portions of said memory; and

a fourth module for testing the integrity of said calculated software variable over a plurality of software test loops, wherein the testing means comprises means for comparing the at least two copies of said calculated software variable to each other.

20. A vehicle having an electronic control module having a processor and a memory configured to store computer-executable instructions for the processor, wherein the instructions comprise:

a verifying module configured to verify the integrity of calculating and storing hardware;

a calculating module configured to calculate a first software variable using said verified calculating and storing hardware, said calculating module further configured to calculate a second software variable different from said first software variable using said verified calculating and storing hardware;

a storing module configured to store said first and second software variables in first and second portions, respectively, of said verified calculating and storing hardware; and

a comparing module configured to compare said first software variable with said second software variable for mutual verification over a plurality of software loops.